We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

4,800
Open access books available

122,000
International authors and editors

135M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

CLARIVATE ANALYTICS BOOK CITATION INDEX INDEXED

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

11

Neural and Adaptive Control Strategies for a Rigid Link Manipulator

Dorin Popescu, Dan Selişteanu, Cosmin Ionete,
Monica Roman and Livia Popescu
Department of Automation and Mechatronics,
University of CraiovaRomania

1. Introduction

The control of robotic manipulators has become important due to the development of the flexible automation. Requirements such as the high speed and high precision trajectory tracking make the modern control indispensable for versatile applications of manipulators (Middleton & Goodwin, 1998; Ortega & Spong, 1999; Popescu *et al.*, 2008). Rigid robot systems are subjects of the research in both robotic and control fields. The reported research leads to a variety of control methods for such rigid robot systems (Ortega & Spong, 1999; Raimondi *et al.*, 2004; Bobaşu & Popescu, 2006; Dinh *et al.*, 2008).

Conventional controllers for robotic structures are based on independent control schemes in which each joint is controlled separately by a simple servo loop. This classical control scheme (for example a PD control) is inadequate for precise trajectory tracking. The imposed performance for industrial applications requires the consideration of the complete dynamics of the manipulator. Moreover, in real-time applications, the ignoring parts of the robot dynamics or errors in the parameters of the robotic manipulator may cause the inefficiency of this classical control. An alternative solution to PD control is the computed torque technique. This classical method is in fact a nonlinear technique that takes account of the dynamic coupling between the robot links. The main disadvantage of this structure is the assumption of an exactly known dynamic model. However, the basic idea of this method remains important and it is the base of the neural and adaptive control structures (Gupta & Rao, 1994; Pham & Oh, 1994; Dumbravă & Olah, 1997; Ortega & Spong, 1999; Aoughellanet *et al.*, 2005; Popescu *et al.* 2008).

Industrial robotic manipulators are exposed to structured and unstructured uncertainties. Structured uncertainties are characterized by having a correct model but with parameter uncertainty (unknown loads and friction coefficients, imprecision of the manipulator link properties, etc.). Unstructured uncertainties are characterized by unmodelled dynamics. Generally speaking, two classes of strategies have been developed to maintain performance in the presence of the parameter uncertainties: robust control and adaptive control. The adaptive controllers can provide good performances in face of very large load variation. Therefore the adaptive approach is intuitively superior to robust approach in this type of application. When the dynamic model of the system is not known a priori (or is not

available), a control law is designed based on an estimated model. This is the basic idea behind adaptive control strategies (Ortega & Spong, 1999).
Over the last few years several authors (Zalzala & Morris, 1996; Miyamoto *et al.*, 1998; Popescu *et al.* 2001; Raimondi *et al.*, 2004; Popescu *et al.*, 2008) have considered the use of artificial neural networks (ANNs) within a control system for robotic arms. The differences in control schemes are in the role that ANN is playing, and the way it is trained for achieving desired trajectory tracking performance.
In this chapter, which is an extended work of the research achieved in some papers of the authors (Popescu, 1998; Popescu *et al.*, 2001, Selişteanu *et al.*, 2001; Popescu *et al.*, 2008), classical, adaptive and neural strategies for a robotic manipulator with two revolute joints are presented. The first section analyses the computed-torque method (based on the so-called inverse dynamics of the robotic manipulator), which is a starting point for the design of the adaptive and neural control techniques. In the next section, an overview of adaptive strategies is presented, and two adaptive controllers for rigid manipulators are designed. First, a direct adaptive control with adaptation law of gradient type is analyzed. Second, an indirect adaptive controller is designed; this controller uses the prediction errors of the filtered joint torques to generate parameter estimates. In the following section, various non-model and model-based neural control schemes have been designed. The ANN is used in order to generate auxiliary joint control torque to compensate for the uncertainties in the computed torque based primary robotic manipulator. Three neural control strategies are studied: feedforward neural control, feedback neural control, and feedback error based neural control. Also, numerical simulations are performed, in order to analyse the behaviour and the performance of the control strategies, and to make some useful comparisons. The final section deals with concluding remarks and further research directions.

2. The computed-torque control strategy

The robotic manipulator is modeled as a set of n rigid bodies connected in series with one end fixed to the ground and the other end free. The bodies are connected via either revolute or prismatic joints and a torque actuator acts at each joint.
The dynamic equation of an n-link robotic manipulator is given by (Ivănescu, 2003; Popescu, 1998):

$$T = J(q)\ddot{q} + V(q,\dot{q})\dot{q} + G(q) + F(\dot{q}), \tag{1}$$

where: T is an $(n\times 1)$ vector of joint torques; $J(q)$ is the $(n\times n)$ manipulator inertia matrix; $V(q,\dot{q})$ is an $(n\times n)$ matrix representing centrifugal and Coriolis effects; $G(q)$ is an $(n\times 1)$ vector representing gravity; $F(\dot{q})$ is an $(n\times 1)$ vector representing friction forces; $q,\dot{q},\ddot{q}$ are the $(n\times 1)$ vectors of joint positions, speeds and accelerations, respectively.
The equations (1) form a set of coupled nonlinear ordinary differential equations which are quite complex, even for simple robotic arms. For simplicity, we denote

$$V(q,\dot{q})\dot{q} + G(q) + F(\dot{q}) = H(q,\dot{q}), \tag{2}$$

so that (1) can be rewritten as:

$$T = J(q)\ddot{q} + H(q,\dot{q}). \tag{3}$$

The computed-torque method is a conventional control technique, which takes account of the dynamic coupling between the manipulator links. This method, also called the inverse model control technique (Zalzala & Morris, 1996; Ortega & Spong, 1999) leads to a completely decoupled error dynamics equation. The structure of this control strategy is illustrated in Fig. 1.

One of most used computed-torque control scheme is based on the exactly linearization of the nonlinear dynamics of the robotic manipulator. If the dynamic model is exact, the dynamic perturbations are exactly cancelled. The total torque driving the robotic manipulator is given by (Dumbravă & Olah, 1997):

$$T = \hat{J}(q)T' + \hat{V}(q,\dot{q})\dot{q} + \hat{G}(q) + \hat{F}(\dot{q}) = \hat{J}(q)T' + \hat{H}(q,\dot{q}), \tag{4}$$

where: $\hat{J}, \hat{V}, \hat{G}, \hat{F}, \hat{H}$ are estimates of J, V, G, F, H, respectively, and T' is defined as:

$$T' = \ddot{q}_d + K_V\dot{e} + K_P e. \tag{5}$$

The closed loop equation is found to be:

$$\ddot{e} + K_V\dot{e} + K_P e = \hat{J}^{-1}(q)\left[\tilde{J}(q)\ddot{q} + \tilde{V}(q,\dot{q})\dot{q} + \tilde{G}(q) + \tilde{F}(\dot{q})\right] = \hat{J}^{-1}(q)\left[\tilde{J}(q)\ddot{q} + \tilde{H}(q,\dot{q})\right], \tag{6}$$

where $\tilde{J} = J - \hat{J};\ \tilde{V} = V - \hat{V};\ \tilde{G} = G - \hat{G};\ \tilde{F} = F - \hat{F};\ \tilde{H} = H - \hat{H}$ are the modelling errors and the tracking error is $e = q_d - q$.

If the robotic manipulator's parameters are perfectly known, the closed loop equation (6) takes a linear, decoupled form:

$$\ddot{e} + K_V\dot{e} + K_P e = 0. \tag{7}$$

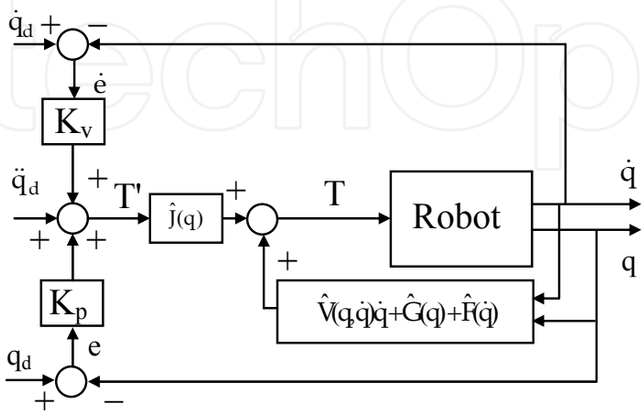


Fig. 1. The computed-torque control scheme

The computed-torque control method has performance problems because of its reliance on a fixed dynamic model. The robotic arm structures have to face uncertainty in the dynamics parameters. Two classes of approach have been studied to maintain performances in the presence of parametric uncertainties - the robust control and the adaptive control. The next section deals with the adaptive control strategy for the robotic manipulator.

3. The adaptive control method

Adaptive controllers can be a good alternative when it is neither possible nor economical to make a thorough investigation of the causes of the process variations. In other situations, some of the dynamics may be well understood, but other parts are unknown. This is the case of robots, for which the geometry, motors and gearboxes do not change, but the load does change. An adaptive controller can be defined as a controller with adjustable parameters and a mechanism for adjusting the parameters (Astrom & Wittenmark, 1995). The modern adaptive control approach consists in the explicit introduction of the linear parameterization of the robot dynamics. The adaptive controllers can be classified into three major categories (Zalzala & Morris, 1996): direct, indirect and composite.

3.1 Direct adaptive controller

The direct adaptive controllers use tracking errors of the joint motion to drive parameter adaptation. The main goal of the control strategy is to reduce the tracking errors. Such a direct technique is an adaptive control method based on computed torque control. This method has been pioneered by (Craig *et al.*, 1987), and the properties of stability and convergence are established in (Slotine & Li, 1987; Ortega & Spong, 1999). The controller is in fact composed of a modified computed-torque control and an adaptation law.

Next, this direct adaptive strategy is used for the robot arm structure (1). Let's consider θ the vector of the uncertain (unknown) parameters, which are the viscous friction coefficients, the Coulomb friction coefficients and the load mass. Then, the dynamics of the robot arm can be written as:

$$T = J(q,\theta)\ddot{q} + V(q,\dot{q},\theta)\dot{q} + G(q,\theta) + F(\dot{q},\theta). \tag{8}$$

A linear parameterization of (8) is:

$$J(q,\theta)\ddot{q} + V(q,\dot{q},\theta)\dot{q} + G(q,\theta) + F(\dot{q},\theta) = J_C(q)\ddot{q} + V_C(q,\dot{q})\dot{q} + G_C(q) + F_C(\dot{q}) + R(q,\dot{q},\ddot{q})\theta, \tag{9}$$

where $J_C(.), V_C(.), G_C(.), F_C(.)$ represent the known (certain) part of the dynamics and $R(q,\dot{q},\ddot{q})$ is the regressor matrix.

The design of the control law is reached by using in (8) the vector of the estimated parameters. The linearization (9) allows us to obtain the torque:

$$T = J(q,\hat{\theta})\ddot{q} + V(q,\dot{q},\hat{\theta})\dot{q} + G(q,\hat{\theta}) + F(\dot{q},\hat{\theta}) = J_C(q)\ddot{q} + V_C(q,\dot{q})\dot{q} + G_C(q) + F_C(\dot{q}) + R(q,\dot{q},\ddot{q})\hat{\theta}, \tag{10}$$

where $\hat{\theta}$ is the vector of estimated parameters.

From the equations (5), (10) the closed loop dynamics is obtained:

$$J(q,\hat{\theta})(\ddot{e}+K_v\dot{e}+K_p e)=R(q,\dot{q},\ddot{q})\tilde{\theta}\,, \tag{11}$$

with $\tilde{\theta}\overset{\Delta}{=}\hat{\theta}-\theta$ the estimation parameter error vector.
If the inertia matrix is nonsingular, we can write:

$$\ddot{e}+K_v\dot{e}+K_p e=J^{-1}(q,\hat{\theta})R(q,\dot{q},\ddot{q})\tilde{\theta}\,. \tag{12}$$

The state representation of (12) can be obtained if the state $x\overset{\Delta}{=}[e\ \ \dot{e}]^T$ is used:

$$\dot{x}=A_m x+B_m J^{-1}(q,\hat{\theta})R(q,\dot{q},\ddot{q})\tilde{\theta}\,, \tag{13}$$

where $A_m=\begin{bmatrix}0 & I\\ -K_p & -K_v\end{bmatrix}$, $B_m=\begin{bmatrix}0\\ I\end{bmatrix}$.
We can choose a gradient type adaptation law for the on-line estimation of the parameters:

$$\frac{d\tilde{\theta}(t)}{dt}=\frac{d\hat{\theta}(t)}{dt}=-\Omega\cdot R^T(q,\dot{q},\ddot{q})J^{-1}(q,\hat{\theta})\cdot B_m^T P x\,, \tag{14}$$

with $\Omega=\Omega^T>0$ the amplification matrix and $P=P^T>0$ a quadratic $n\times n$ matrix, solution of the Lyapunov equation:

$$A_m^T P+PA_m=-Q\,, \tag{15}$$

where $Q=Q^T>0$.

Remark 1. The Lyapunov function $V=x^T Px+\tilde{\theta}^T\tilde{\theta}$ can be used to show that the tracking errors go to zero. □

The final adaptive control law consists of the computed-torque Eq. (4) and the estimates provided by the adaptation law (14).
The global convergence of the direct adaptive controller based on computed-torque method is demonstrated in (Slotine & Li, 1987). The disadvantages of this adaptive method are the use of the acceleration measurements and the necessity of inversion of the estimated inertia matrix. The advantages are the simplicity of the method (comparatively to a least squares indirect method for example) and the rejection of the parametric disturbances, inherent for an adaptive method.

3.2 Indirect adaptive controller

The indirect adaptive control method for manipulators has been pioneered by (Middleton & Goodwin, 1998), who used prediction errors on the filtered joint torques to generate parameter estimates to be used in the control law.

Such indirect adaptive controller can be composed of a modified computed-torque control and a modified least-squares estimator.
The design of this indirect control law for the manipulator (1) is based on the estimate of the torque:

$$\hat{T} = J_C(q)\ddot{q} + V_C(q,\dot{q})\dot{q} + G_C(q) + F_C(\dot{q}) + R(q,\dot{q},\ddot{q})\hat{\theta}\,, \tag{16}$$

where $\hat{\theta}$ is the vector of estimated parameters.
Now we can calculate the prediction error for the torque from (9), (16)

$$\varepsilon \overset{\Delta}{=} \hat{T} - T = R(q,\dot{q},\ddot{q})\cdot(\hat{\theta} - \theta) = R(q,\dot{q},\ddot{q})\tilde{\theta}\,, \tag{17}$$

with $\tilde{\theta} \overset{\Delta}{=} \hat{\theta} - \theta$ the estimation parameter error vector.
The prediction error is filtered to eliminate the measurements of the accelerations in the control law. First, the torque T is filtered through a first-degree filter with the transfer function $H(s) = \dfrac{\omega_f}{s + \omega_f}$, where ω_f is the crossover frequency of the filter. The filtered torque is the convolution

$$T_f = h(t) * T(t)\,, \tag{18}$$

where h(t) is the impulse response of H(s).
The estimated torque is also filtered. We define

$$T_C = J_C(q)\ddot{q} + V_C(q,\dot{q})\dot{q} + G_C(q) + F_C(\dot{q})\,, \tag{19}$$

and from (16), (19) the estimated torque can be written as

$$\hat{T} = T_C + R(q,\dot{q},\ddot{q})\hat{\theta}\,. \tag{20}$$

We have

$$T_{Cf}(t) = h(t) * T_C(t)\,, \tag{21}$$

$$\Phi(t) = h(t) * R(t)\,. \tag{22}$$

In the relations (21), (22), T_{Cf} and the filtered regressor matrix Φ depend only of the state q(t) and of the time derivative $\dot{q}(t)$, and not of the accelerations (Ivănescu, 2003):

$$T_{Cf}(t) = T_{Cf}(q(t),\dot{q}(t)); \quad \Phi(t) = \Phi(q(t),\dot{q}(t))\,. \tag{23}$$

We obtain the filtered estimated torque from (20), (21), and (22):

$$\hat{T}_f = T_{Cf} + \Phi \cdot \hat{\theta} \,. \tag{24}$$

Now we can obtain the filtered prediction error, which will be used in the adaptation law. From (17), (18), (24) the filtered prediction error is

$$\varepsilon_f = \hat{T}_f - T_f = T_{Cf}(t) + \Phi(t) \cdot \hat{\theta} - h(t) * T(t) \,. \tag{25}$$

The torque T can be written as

$$T = T_C + R \cdot \theta \,, \tag{26}$$

therefore the filtered prediction error becomes

$$\varepsilon_f = h(t) * T_C(t) + \Phi(t) \cdot \hat{\theta} - h(t) * T_C(t) - \Phi(t) \cdot \theta = \Phi(q(t), \dot{q}(t)) \cdot \tilde{\theta} \,. \tag{27}$$

The adaptation parameter law is based on a least-squares estimator (Dumbravă & Olah, 1997) that it has as input the filtered prediction error (27). The equations of the adaptation law are

$$\frac{d\tilde{\theta}(t)}{dt} = \frac{d\hat{\theta}(t)}{dt} = -\Gamma(t)\Phi^T(q,\dot{q})\varepsilon_f(t) \,, \tag{28}$$

$$\frac{d\Gamma(t)}{dt} = -\Gamma(t)\Phi^T(q,\dot{q})\Phi(q,\dot{q})\Gamma^T(t) \,, \tag{29}$$

with $\Gamma(0) = \Gamma^T(0) > 0.$ The matrix $\Gamma(t) = \Gamma^T(t) > 0$ is the amplification matrix.
The final indirect adaptive control law consists of the computed-torque equation (4) and the estimates provided by the adaptation law (28), (29):

$$T = \hat{J}(q)T' + \hat{V}(q,\dot{q})\dot{q} + \hat{G}(q) + \hat{F}(\dot{q}) = J(q,\hat{\theta})T' + V(q,\dot{q},\hat{\theta})\dot{q} + G(q,\hat{\theta}) + F(\dot{q},\hat{\theta}) \,, \tag{30}$$

with T' given by (5).
The indirect adaptive control structure is presented in Fig. 2.
The least-squares estimator (28), (29) has good convergence and stability properties (Ivănescu, 2003). A disadvantage can be the complexity of the algorithm and the correlation between the prediction error and the estimation parameter error (Ivănescu, 2003; Dumbravă & Olah, 1997). This disadvantage can be canceled by addition of a stabilizing signal to the control law (Dumbravă & Olah, 1997).

Remark 2. Indirect controllers allow the various parameter-estimation algorithms to be used to select time variations of the adaptation gains. □

Composite adaptive controllers for manipulators have been developed by (Slotine & Li, 1989). These adaptive control strategies use both tracking errors in the joint motions and prediction errors on the filtered torque to drive the parameter adaptation.

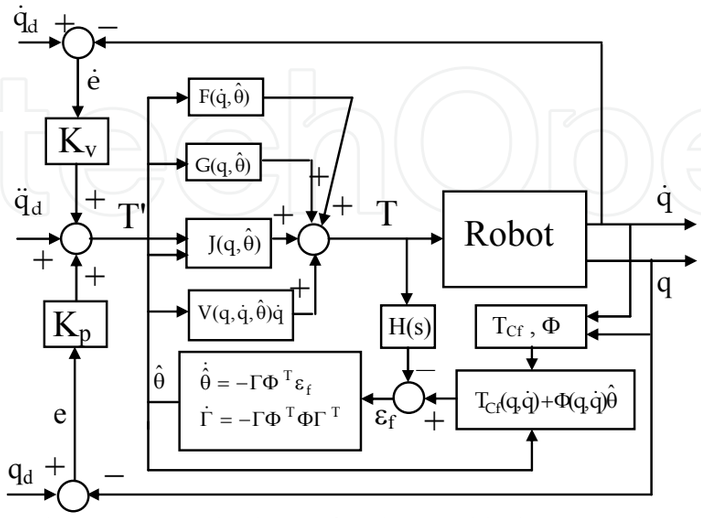


Fig. 2. The indirect adaptive control scheme for the rigid link manipulator

4. Neural control strategies

Various neural control schemes have been studied, proposed and compared. The differences in these schemes are in the role that artificial neural network (ANN) is playing in the control system and the way it is trained for achieving desired trajectory tracking performance structures (Psaltis *et al.*, 1988; Gupta & Rao, 1994; Zalzala & Morris, 1996; Raimondi *et al.*, 2004; Aoughellanet *et al.*, 2005; Dinh *et al.*, 2008). Two classes of approaches have been studied: non-model based neural control and model based neural control. Non-model based neural control consists of PD feedback controller and an ANN. The inverse dynamics is learned by measuring the input and output signals in the manipulator and then adjusting the connection weights vector by using a learning algorithm. After the learning was finished, the actual trajectory of the manipulator followed well the desired trajectory. But, when the desired trajectory was changed to one not used in the training of ANN, the error between the actual and desired trajectory became large. This means that the ANN had fitted a relationship between the input/output data but had no succeeded in learning the inverse-dynamics model (Zalzala & Morris, 1996). We want that training doesn't depend on desired trajectory. Hence, we proposed to train the ANN with $(q,\dot{q},\ddot{q}_d,e,\dot{e})$ (see Fig. 3).

For training of ANN, there are two possibilities: off-line or on-line. From the viewpoint of real time control it's better to train ANN on-line. But, from the viewpoint of initial weights and biases, rate of convergence and stability of learning it's better to train ANN off-line. The tracking performance was better if ANN was trained off-line and then ANN was used to improve the performance of PD feedback controller (Popescu *et al.*, 2001).

In this section, model based neural control structures for a robotic manipulator are implemented. Various neural control schemes have been studied, proposed and compared. The differences in these schemes are in the role that ANN is playing in the control system

and the way it is trained for achieving desired trajectory tracking performance. The most popular control scheme is one which uses ANN to generate auxiliary joint control torque to compensate for the uncertainties in the computed torque based primary robotic manipulator controller that is designed based on a nominal robotic manipulator dynamic model.
This is accomplished by implementing the neural controller in either a feedforward or a feedback configuration, and the ANN is trained on-line. Based on the computed torque method, a training signal is derived for neural controller. Comparison studies based on a robotic planar manipulator have been made for the neural controller implemented in both feedforward and feedback configurations. Also, a feedback error based neural controller is proposed. In this approach, a feedback error function is minimized and the advantage over Jacobian based approach is that Jacobian estimation is not required.

4.1 Feedforward neural control strategy

The feedforward neural controller (Fig. 3) is designed to achieve perturbation rejection for a computed torque control system of a robotic manipulator. The ANN output cancels out the uncertainties caused by inaccurate robotic manipulator's model in the computed torque controller. The robot joint torques are:

$$T = \hat{J}(q_d)(T' + \phi_f) + \hat{H}(q_d, \dot{q}_d)\,. \tag{31}$$

The closed loop error system is

$$\ddot{e} + K_V \dot{e} + K_P e = \hat{J}^{-1}\left(\tilde{J}\ddot{q} + \tilde{H}\right) - \phi_f\,. \tag{32}$$

Since the control objective is to generate ϕ_f to reduce u to zero, we therefore propose to use:

$$u = \ddot{e} + K_V \dot{e} + K_P e\,, \tag{33}$$

as the error signal for training the ANN. Then, the ideal value of ϕ_f at u = 0 is:

$$\phi_f = \hat{J}^{-1}\left(\tilde{J}\ddot{q} + \tilde{H}\right). \tag{34}$$

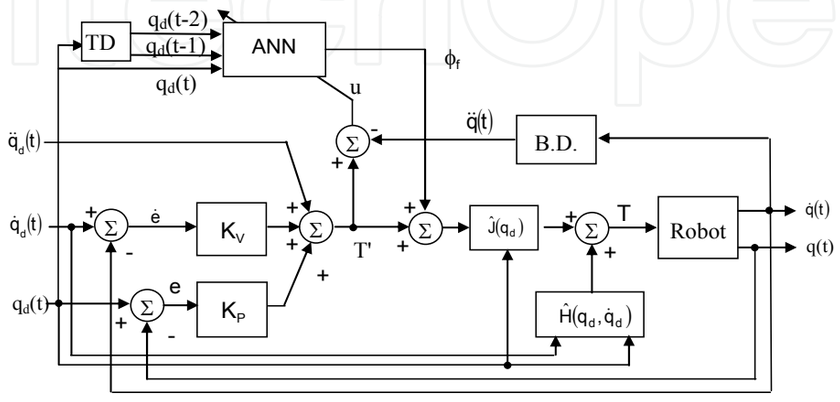


Fig. 3. The structure of the feedforward neural controller

4.2 Feedback neural control strategy

The main difference between feedforward and feedback neural controller schemes is that the joint variables used in the ANN inputs and the computed torque controller are either the desired values $q_d(t)$ or the actual values $q(t)$. The ANN inputs can be either $q_d(t)$, $\dot{q}_d(t)$, $\ddot{q}_d(t)$, or $q(t)$, $\dot{q}(t)$, $\ddot{q}(t)$, or the time-delayed values $q_d(t)$, $q_d(t-1)$, $q_d(t-2)$, or $q(t)$, $q(t-1)$, $q(t-2)$. Delay time is chosen as the sampling period of the controller. In simulations the ANN performs better when time-delayed joint values are used instead of the velocity and acceleration values calculated from finite difference approximations based on samples of $q(t)$.

For feedback neural controller (Fig. 4) the robotic manipulator joint torques are:

$$T = \hat{J}(q)(T' + \phi_b) + \hat{H}(q,\dot{q}) . \tag{35}$$

The three-layer feedforward neural network is used as the compensator. It is composed of an input layer (6 neurons), a nonlinear hidden layer, and a linear output layer (2 neurons). The weight updating law minimizes the objective function J which is a quadratic function of the training signal u:

$$J = \frac{1}{2}\left(u^T u\right). \tag{36}$$

For simplicity, we use ϕ for ϕ_f or ϕ_b. Differentiating equation (36) and making use of (32) yields the gradient of J as follows:

$$\frac{\partial J}{\partial w} = \frac{\partial u^T}{\partial w} u = -\frac{\partial \phi^T}{\partial w} u . \tag{37}$$

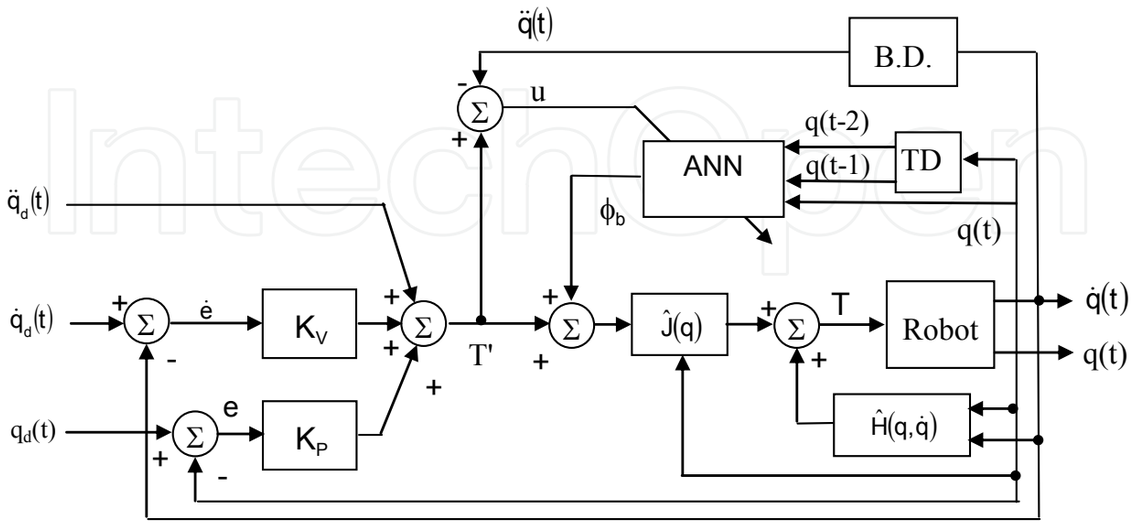


Fig. 4. The structure of the feedback neural controller

The backpropagation update rule for the weights with a momentum term is:

$$\Delta w(t) = -\eta \frac{\partial J}{\partial w} + \alpha \Delta w(t-1) = \eta \frac{\partial \phi^T}{\partial w} u + \alpha \Delta w(t-1), \tag{38}$$

where η is the learning rate and α is the momentum coefficient.

4.3 Feedback error based neural control strategy

In this approach, a feedback error function is minimized and the advantage over Jacobian based approach is that Jacobian estimation is not required. The inputs to the neural controller (Fig. 5) are the required trajectories $q_d(t)$, $\dot{q}_d(t)$, $\ddot{q}_d(t)$. The compensating signals from ANN, ϕ_p, ϕ_v, ϕ_a, are added to the desired trajectories.

The control law is:

$$T = \hat{J}\left(\ddot{q}_d + \phi_a + K_V(\dot{e} + \phi_v) + K_P(e + \phi_p)\right) + \hat{H}. \tag{39}$$

Combining (39) with dynamic equation of robotic manipulator yields:

$$u = \ddot{e} + K_V \dot{e} + K_P e = \hat{J}^{-1}\left(\tilde{J}\ddot{q} + \tilde{H}\right) - \Phi, \tag{40}$$

where $\Phi = \phi_a + K_V \phi_v + K_P \phi_p$. Ideally, at u = 0, the ideal value of Φ is:

$$\Phi = \hat{J}^{-1}\left(\tilde{J}\ddot{q} + \tilde{H}\right). \tag{41}$$

The error function u is minimized and objective function is the same – see expression (36).

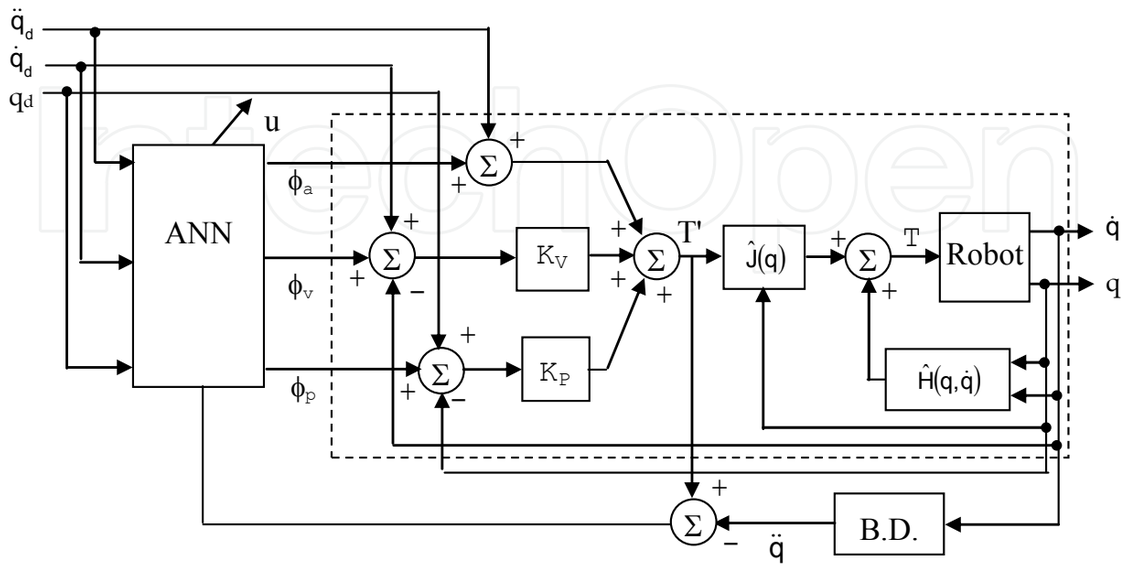


Fig. 5. The structure of the feedback error based neural controller

The gradient of J is:

$$\frac{\partial J}{\partial w} = \frac{\partial u^T}{\partial w} u = -\frac{\partial \Phi^T}{\partial w} u \,. \tag{42}$$

The backpropagation updating rule for the weights with momentum term is:

$$\Delta w(t) = -\eta \frac{\partial J}{\partial w} + \alpha \Delta w(t-1) = \eta \frac{\partial \Phi^T}{\partial w} u + \alpha \Delta w(t-1) \,. \tag{43}$$

5. Simulation results and comparisons

In order to test the proposed adaptive and neural control strategies, the control of the simple planar robotic manipulator with two revolute joints shown in Fig. 6 was considered.
The elements of the dynamic equation (1) for this robotic manipulator with electrical motor dynamics are (Selişteanu *et al.*, 2001; Popescu *et al.*, 2008):

$$J(q) = \begin{bmatrix} l_1^2(m_1+m_2) + m_2 l_2^2 + 2m_2 l_1 l_2 c_2 + J_1 n_1^2 & m_2 l_2^2 + m_2 l_1 l_2 c_2 \\ m_2 l_2^2 + m_2 l_1 l_2 c_2 & m_2 l_2^2 + J_2 n_2^2 \end{bmatrix}, \tag{44}$$

$$V(q,\dot{q}) = m_2 l_1 l_2 s_2 \begin{bmatrix} 0 & -(2\dot{q}_1 + \dot{q}_2) \\ \dot{q}_1 & 0 \end{bmatrix}, \tag{45}$$

$$G(q) = \begin{bmatrix} (m_1+m_2) g l_1 s_1 + m_2 g l_2 s_{12} \\ m_2 g l_2 s_{12} \end{bmatrix}, \tag{46}$$

$$F(\dot{q}) = \begin{bmatrix} v_1 \dot{q}_1 + C_1 \mathrm{sign}(\dot{q}_1) \\ v_2 \dot{q}_2 + C_2 \mathrm{sign}(\dot{q}_2) \end{bmatrix}, \tag{47}$$

with m_1 - mass of link 1, $m_2 = m_{20} + m_p$, m_{20} - mass of link 2, m_p - mass of payload, l_1 - length of link 1, l_2 - length of link 2, $c_i = \cos(q_i), i = 1,2$, $s_i = \sin(q_i), i = 1,2$, $c_{12} = \cos(q_1 + q_2)$, $s_{12} = \sin(q_1 + q_2)$, $J_i, i = 1,2$ - moments of inertia for electrical motor i, $n_i, i = 1,2$ - factor of reduction gear i, $v_i, i = 1,2$ - viscous friction for joint i, $C_i, i = 1,2$ - Coulomb friction for joint i.

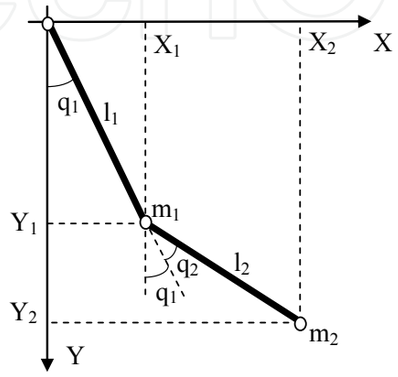


Fig. 6. The planar robotic manipulator with two revolute joints

For simulation and comparisons, the planar robot manipulator with two revolute joints (1), (44)-(47) is used. The simulation model parameters are (SI units): $m_1 = 10$, m_2 consists of mass of link 2, $m_{20} = 2.5$, and mass of payload m_p, $l_1 = 1, l_2 = 0.5$, link lengths.

The robot manipulator starts at position $(q_1 = 0, q_2 = 0)$, and the control objective is to track the desired trajectory given by:

$$q_{1d} = 0.4 \cdot \sin(0{,}4\pi t),\ q_{2d} = -0.5 \cdot \sin(0{,}5\pi t). \tag{48}$$

In order to test the performance and to analyze the behaviour of the control strategies, several numerical simulations were performed. We considered three basic simulation cases:

1) When the model of robot manipulator is known, the use of the computed-torque method is recommended. The equations (4), (5) are used and a simulation has been done for the tuning parameters $K_{p1} = 50, K_{p2} = 50, K_{v1} = 6, K_{v2} = 15$ (matrices K_p, K_v of diagonal form). The time evolution of tracking errors $e = [e_1 \ \ e_2]^T = [q_{1d} - q_1 \ \ q_{2d} - q_2]^T$ is presented in Fig. 7.

2) The computed-torque method provides good results when the model is exactly known. If parametric uncertainties occur, an adaptive control method can be used. Let's consider that the uncertain parameters are the viscous friction coefficients, the Coulomb friction coefficients and the load mass. Therefore we have: $\theta = [m_p \ v_1 \ C_1 \ v_2 \ C_2]^T$. The direct adaptive control law (9), (10), (14) is implemented with the design parameters $K_{p1} = 50,\ K_{p2} = 50,\ K_{v1} = 6,\ K_{v2} = 15,$ and the diagonal matrix $\Omega = [\omega_{ii}]_{i=\overline{1,5}},\ \omega_{ii} = 15$. The results are presented in Fig. 8.

We can see that even if the estimated parameters $\hat{\theta} = [\hat{m}_p \ \hat{v}_1 \ \hat{C}_1 \ \hat{v}_2 \ \hat{C}_2]^T$ are used, the evolution of tracking errors remains good.

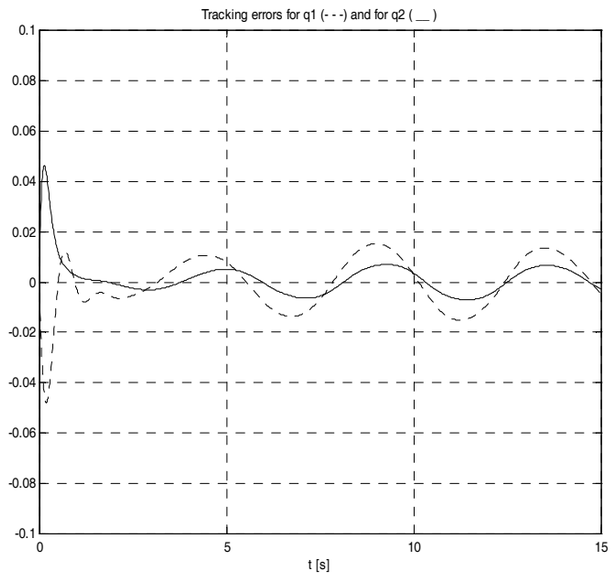


Fig. 7. Tracking errors – the computed-torque case

The imposed trajectories are preserved in the presence of the parametric uncertainties.
Another simulation is done for the indirect adaptive control law (28)-(30), which is implemented with the parameters $K_{p1}=50,\ K_{p2}=50,\ K_{v1}=6,\ K_{v2}=15,\ \omega_f=5$ and the diagonal matrix $\Gamma(0)=[\gamma_{ii}]_{i=\overline{1,5}},\ \gamma_{ii}=15$. The estimated parameters have a fast convergence to their actual ("true") values. The evolution of tracking errors is illustrated in Fig. 9.

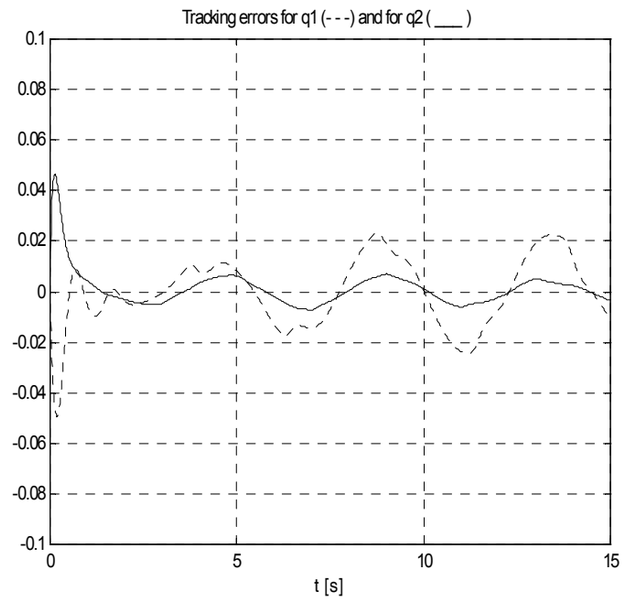


Fig. 8. Simulation results – direct adaptive control law

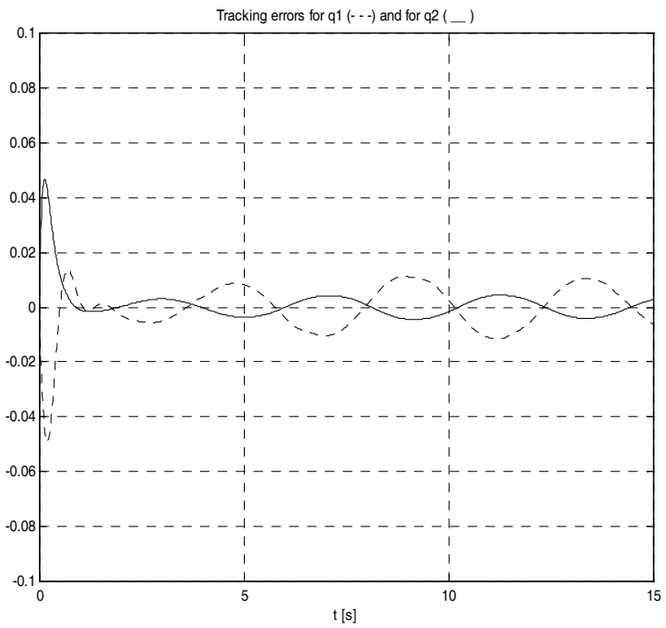


Fig. 9. Tracking errors – the indirect adaptive controller

3) The neural control strategies were implemented. For feedforward neural control law and for feedback neural controller (with the neural network structure of the form 6 x 10 x 2), with update backpropagation rule (38), the time evolution of the tracking errors is presented in Fig. 10 and Fig. 11, respectively.
Also, for the feedback error based neural control law (with the neural network structure of the form 6 x 9 x 2), and with the update backpropagation rule (43), the time profiles of the tracking errors are depicted in Fig. 12.

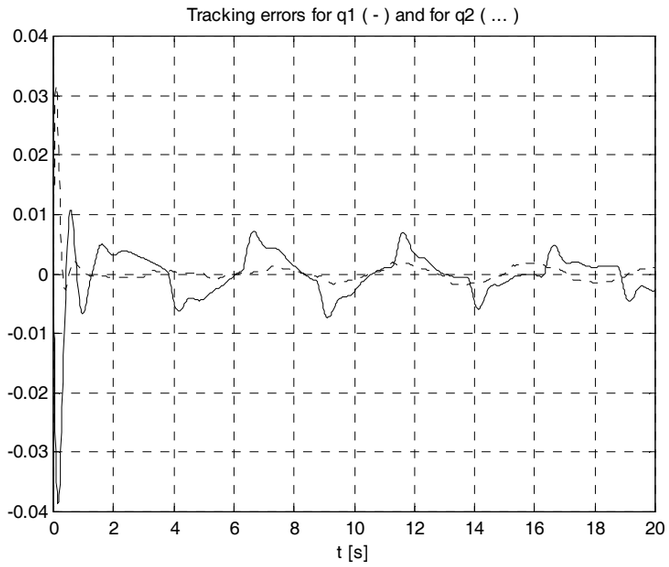


Fig. 10. Tracking errors – the feedforward neural control scheme

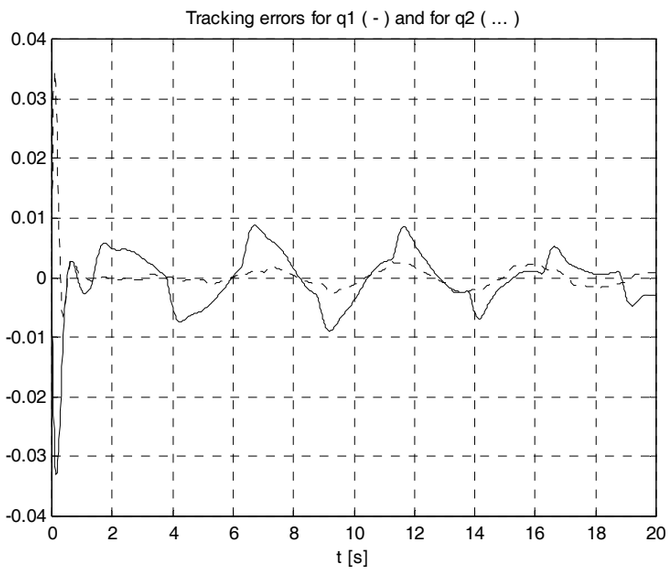


Fig. 11. Simulation results – the feedback neural controller

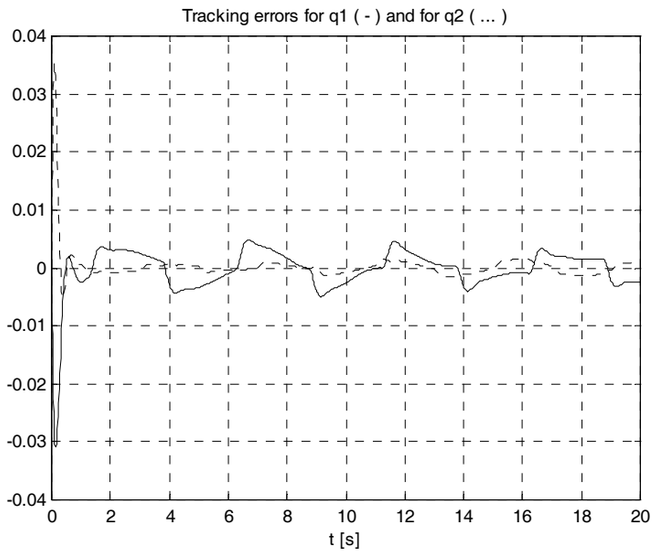


Fig. 12. Tracking errors – the feedback error based neural control scheme

The comparisons between the control strategies can be done by visualization of tracking errors, but accurate comparisons can be done by considering a criterion based on averaged square tracking errors – see (Whitcomb *et al.*, 1991; Popescu *et al.*, 2008):

$$I_1 = \frac{1}{T_S}\int_0^{T_S} e_1^2(t)dt\,,\quad I_2 = \frac{1}{T_S}\int_0^{T_S} e_2^2(t)dt\,, \tag{49}$$

where T_S is the total simulation time.

The values of I_1 and I_2, computed for the studied control strategies (manipulator with two revolute joints), and for numerous simulations (including PD control law and non-model based neural controllers) are presented in Table 1.

Control Strategies	Performances	
	I_1	I_2
Classical PD controller (exactly known model and manipulator parameters)	4.2*10-4	3.1*10-4
Computed-torque method (exactly known model and manipulator parameters)	1.2*10-4	0.8*10-4
Direct adaptive controller (gradient adaptation law)	1.7*10-4	1.0*10-4
Indirect adaptive controller (least-squares estimator as adaptation law)	1.1*10-4	0.7*10-4
Non-model based neural controller	2*10-4	1.1*10-4
Model based neural controller	0.9*10-4	0.5*10-4

Table 1. Performance criterion results

6. Conclusion

In this chapter, some classical, adaptive and neural control strategies for a simple planar robotic manipulator with two revolute joints were designed and implemented.
First, the conventional computed-torque method was discussed. This control method solves the precision tracking problem, by using an exactly linearization of the nonlinearities of the manipulator model. The main disadvantage is the assumption of an exactly known dynamic model. If the model is imprecise known, it is necessary to design adaptive and/or neural control strategies.
Direct and indirect adaptive controllers have been studied and implemented in order to preserve the tracking performances when parameter uncertainties occur. From the simulation point of view, it can be noticed that the evolution of tracking errors remains good, even if the estimated parameters are used in the control law.
Also, three neural based control strategies were developed: a feedforward neural controller, a feedback neural control scheme, and a feedback error based neural controller. The simulations showed that the proposed neural controllers obtain results comparable to those achieved using adaptive control strategies. If a classical (PD or computed-torque) controller already controls a manipulator the advantage of proposed neural structures is that extension to a neural controller for performances improvement is easy.

7. Acknowledgment

This work was supported by the National Authority for Scientific Research, Romania, under the research project SICOTIR, 71-084/2007 (PNCDI II).

8. References

Aoughellanet, S.; Mohammedi, T. & Bouterfa, Y. (2005). Neural network path planning applied to PUMA560 robot arm. *WSEAS Transactions on Systems*, Issue 4, Vol. 4, April 2005, pp. 446-450, ISSN: 1109-2777

Astrom, K.J. & Wittenmark, B. (1995). *Adaptive Control*, Addison-Wesley, ISBN: 0-0201-55866-1, USA

Bobaşu, E. & Popescu, D. (2006). On modelling and multivariable adaptive control of robotic manipulators. *WSEAS Transactions on Systems*, Issue 7, Vol. 5, July 2006, pp. 1579-1586, ISSN: 1109-2777

Craig, J.J.; Hsu, P. & Sastry, S. (1987). Adaptive control of mechanical manipulators. *International Journal of Robotics Research*, Vol. 2, pp. 10-20

Dinh, B.H.; Dunnigan, M.W. & Reay, D.S. (2008). A practical approach for position control of a robotic manipulator using a radial basis function network and a simple vision system. *WSEAS Transactions on Systems and Control*, Issue 4, Vol. 3, April 2008, pp. 289-298, ISSN: 1991-8763

Dumbravă, S. & Olah, I. (1997). Robustness analysis of computed torque based robot controllers, *Proceedings of 5-th Symposium on Automatic Control and Computer Science*, pp. 228-233, Iaşi, Romania, 1997

Gupta, M.M. & Rao, D.H. (1994). *Neuro-Control Systems: Theory and Applications*, IEEE Press Piscataway, NJ, USA

Ivănescu, M. (2003). *Advanced Systems for Robotic Control*, (in Romanian), Ed. Scrisul Romanesc, Craiova, ISBN: 973-38-0389-8

Middleton, R. & Goodwin, G. (1998). Adaptive computed torque control for rigid link manipulators. *Systems and Control Letters*, Vol. 10, pp. 9-16, ISSN: 0167-6911

Miyamoto, H.; Kawato, M.; Setoyama, T. & Suzuki, R. (1998). Feedback error learning neural networks for trajectory control of a robotic manipulator. *Neural Networks*, Vol. 1, pp. 251-265, ISSN: 1045-9227

Ortega, R. & Spong, M.W. (1999). Adaptive motion control of rigid robots: a tutorial. *Automatica*, Vol. 25, pp. 877-888, ISSN: 0005-1098

Pham, D.T. & Oh, S.J. (1994). Adaptive control of a robot using neural networks. *Robotica*, pp. 553-561, ISSN: 0263-5747

Popescu, D. (1998). Neural control of manipulators using a supervisory algorithm, *Proceedings of A&Q'98 International Conference on Automation and Quality Control*, pp. A576-A581, Cluj-Napoca, Romania, 1998, Ed. Mediamira, ISBN: 973-9358-15-2

Popescu, D.; Selişteanu, D. & Ionete, C. (2001). Non-model based neural robot control, *Proceedings of the 10th International Workshop on Robotics in Alpe-Adria-Danube Region*, RD-038, Vienna, Austria, 2001

Popescu, D.; Selişteanu, D. & Popescu, L. (2008). Neural and adaptive control of a rigid link manipulator. *WSEAS Transactions on Systems*, Issue 6, Vol. 7, pp. 632-641, ISSN: 1109-2777

Psaltis, D.; Sideris, A. & Yamamura, A. (1988). A multilayered neural network controller. *IEEE Control Systems Magazine*, Vol. 8, pp. 17-21, ISSN: 0272-1708

Raimondi, F.M.; Melluso, M. & Bonafede, V. (2004). A neuro fuzzy controller for planar robot manipulators. *WSEAS Transactions on Systems*, Issue 10, Vol. 3, December 2004, pp. 2991-2996, ISSN: 1109-2777

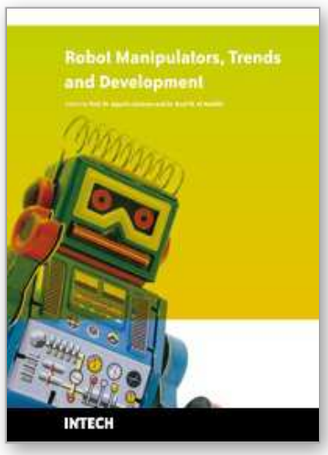
Selişteanu, D.; Popescu, D.; Bîzdoacă, N. & Ionete, C. (2001). Adaptive and neural control of a rigid link manipulator, *Proceedings of the 5th World Multi-Conference on Systemics, Cybernetics and Informatics SCI 2001*, Vol. IX, Part I, pp. 499-504, ISBN 980-07-7549-8, Orlando, SUA, July 2001

Slotine, J.J.E. & Li, W. (1987). On the adaptive control of robot manipulators. *International Journal of Robotics Research*, Vol. 6, Issue 3, pp. 49-59, ISSN: 0278-3649

Slotine, J.J.E. & Li, W. (1989). Composite adaptive manipulator control. *Automatica*, Vol. 25, No. 4, pp. 509-519, ISSN: 0005-1098

Whitcomb, L.L.; Rizzi, A.A. & Kodishek, D.E. (1991). Comparative experiments with a new adaptive controller for robot arms, *Proceedings of IEEE Conference on Robotics & Automation*, pp. 2-7, Sacramento, USA, 1991

Zalzala, A. & Morris, A. (1996). *Neural Networks for Robotic Control*, Prentice Hall, ISBN: 978-0131198920



Robot Manipulators Trends and Development

Edited by Agustin Jimenez and Basil M Al Hadithi

ISBN 978-953-307-073-5
Hard cover, 666 pages
Publisher InTech
Published online 01, March, 2010
Published in print edition March, 2010

This book presents the most recent research advances in robot manipulators. It offers a complete survey to the kinematic and dynamic modelling, simulation, computer vision, software engineering, optimization and design of control algorithms applied for robotic systems. It is devoted for a large scale of applications, such as manufacturing, manipulation, medicine and automation. Several control methods are included such as optimal, adaptive, robust, force, fuzzy and neural network control strategies. The trajectory planning is discussed in details for point-to-point and path motions control. The results in obtained in this book are expected to be of great interest for researchers, engineers, scientists and students, in engineering studies and industrial sectors related to robot modelling, design, control, and application. The book also details theoretical, mathematical and practical requirements for mathematicians and control engineers. It surveys recent techniques in modelling, computer simulation and implementation of advanced and intelligent controllers.

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen